A TESTING ALGORITHM OF AN UNIVERSAL ALGEBRA TO BE A BRANDT GROUPOID

Gheorghe IVAN and George STOIANOV

Abstract.

The main aim of this paper is to present a program on computer for decide if an universal algebra is a groupoid. Using the theory of groupoids and the program $BGroidAP1$ we prove a theorem of classification for the groupoids of type $(4;2)$. [1]

Introduction

The algebraic notion of groupoid was introduced and named by H. Brandt in the paper [*Uber eine Verallgemeinerung der Gruppen-begriffes*. Math. Ann., 96, 1926, 360-366]. A groupoid (in the sense of Brandt) can be thought of as a generalized group in which only certain multiplications are possible and it contains several neutral elements.

Groupoids also appeared in Galois theory in the description of relations between subfields of a field K via morphisms of K in a paper of A. Loewy [*Neue elementare Begrundung und Erweiterung der Galoisschen Theorie*. S.-B. Heidelberger Akad. Wiss. Math. Nat. Kl.**1925**, 1927]. In differentiable context, the concept of groupoid has appeared in the work of C. Ehresmann [*Catégories et structures*. Dunod, Paris] around 1950.

In the language of categories, a groupoid is a small category in which all morphisms are invertible. For more details concerning the groupoids and its applications in many areas of mathematics, see [1] - [4],[6] - [8].

The plan of this paper is as follows. In the first section we have collected the preliminary concepts concerning groupoids. In the second section we present an algorithm for to verify that a finite set endowed with structure functions has a groupoid structure. This algorithm is based on the theory of groupoids and is implemented on computer. The obtained program is denoted by $BGroidAP1$. Finally, we ilustrate the utilisation of the program on some finite universal algebras. In particular, the program can be used for to test if a finite set endowed with a composition law has a structure of group.

1. The concept of Brandt groupoid as universal algebra

Definition 1.1. Let (M, M_0) be a pair of nonempty sets, where M_0 a subset of M endowed with the surjections $\alpha,\ \beta : M \rightarrow M_0$, called the *source* and the *target* map, respectively and a *(partial) multiplication*

[1] AMS classification: 20L13, 68W10.

Key words and phrases: semigroupoid, monoidoid, groupoid, transitive groupoid

law $\mu : M_{(2)} \longrightarrow M, (x,y) \longrightarrow \mu(x,y)$, where $M_{(2)} = \{ (x,y) \in M \times M \mid \beta(x) = \alpha(y) \}$. We write sometimes $x \cdot y$ or xy for $\mu(x,y)$. The elements of $M_{(2)}$ are called *composable pairs* of M.

We say that the universal algebra $(M, \alpha, \beta, \mu; M_0)$ is a *semigroupoid*, if the multiplication law is *associative*, i.e. $(xy)z = x(yz)$, in the sense that, if one side of the equation is defined so is the other one and then they are equal(the element $(xy)z$ is defined iff $\beta(x) = \alpha(y)$ and $\beta(y) = \alpha(z)$). $\square$

Definition 1.2. ([4]) A *monoidoid* is a semigroupoid $(M, \alpha, \beta, \mu; M_0)$ such that the *identities property* holds, i. e. for each $x \in M$ we have $(\alpha(x), x)$, $(x, \beta(x)) \in M_{(2)}$ and $\alpha(x)x = x\beta(x) = x$. $\square$

The element $\alpha(x)$ [resp. $\beta(x)$] denoted sometimes by $u_l(x)$ [resp. $u_r(x)$] is the *left unit* [resp. *right unit*] of $x \in M$. The subset $M_0 = \alpha(M) = \beta(M)$ of M is called the *unit set* of M and we say that M is a M_0 - *monoidoid*. The functions α, β, μ are called *structure functions* of the monoidoid M. For each $u \in M_0$, the set $\alpha^{-1}(u)$ (resp., $\beta^{-1}(u)$) is called α -*fibre* (resp., β -*fibre*) of the monoidoid M over $u \in M_0$.

In the following proposition we summarize some properties of the structure functions of a monoidoid.

Proposition 1.1. ([4]) *Let* $(M, \alpha, \beta, \mu; M_0)$ *and* $u, v \in M_0$. *Then the following assertions hold:*

(1.1) $\alpha(u) = \beta(u) = u$ *and* $u \cdot u = u$ *for all* $u \in M_0$;

(1.2) $\alpha(xy) = \alpha(x)$ *and* $\beta(xy) = \beta(y)$, $(\forall)$ $(x,y) \in M_{(2)}$;

(1.3) *if* $(x,u) \in M_{(2)}$ *such that* $xu = x$ *then* $u = \beta(x)$;

(1.4) *if* $(v,x) \in M_{(2)}$ *such that* $vx = x$ *then* $v = \alpha(x)$; $\square$

Example 1.1. (i) A monoid $\mathcal{M}$ having e as unity, is just a $\{e\}$ - monoidoid over a one point-base in the following way: $M = \mathcal{M}, M_0 = \{e\}$, the source and target maps $\alpha, \beta : M \rightarrow M_0$ are constant maps, i.e. $\alpha(x) = \beta(x) = e$ for all $x \in M$; for all $x, y \in M$ we have $\alpha(x) = \beta(x) = e$ and hence the product $x \cdot y$ is always defined in M ($x \cdot y$ is the product of elements x and y in the monoid $\mathcal{M}$).

Conversely, every monoidoid M with one unit (i.e. M_0 is a singleton) is a monoid.

(ii) The *nul monoidoid over a set*. Any nonempty set X may be regarded as a monoidoid on itself with the following monoidoid structure : $M = M_0 = X, \alpha = \beta = Id_X$; the elements $x, y \in X$ are composable iff $x = y$ and we define $x \cdot x = x$.

(iii) *Disjoint union of two monoidoids*. Let $(M_i, \alpha_i, \beta_i, \mu_i; M_{0,i})$ $, i = 1, 2$ be two monoidoids such that $M_1 \cap M_2 = \emptyset$. We consider $M = M_1 \cup M_2$ and $M_{(2)} = M_{1,(2)} \cup M_{2,(2)}$.

We give on the set M the following structure:

$\alpha(x) = \alpha_1(x)$ if $x \in M_1$ and $\alpha(x) = \alpha_2(x)$ if $x \in M_2$;

$\beta(x) = \beta_1(x)$ if $x \in M_1$ and $\beta(x) = \beta_2(x)$ if $x \in M_2$;

we have that $(x,y) \in M_{(2)}$ iff $(x,y) \in M_{1,(2)}$ or $(x,y) \in M_{2,(2)}$ and we take $\mu(x,y) = \mu_i(x,y)$ if $(x,y) \in M_{i,(2)}$ $, i = 1, 2$.

In other words, two elements $x, y \in M$ may be composed iff they lie in the same monoidoid $M_i, i=1,2$ and they are composable in $M_i, i=1,2$.

In the case when $M_1 \cap M_2 \neq \emptyset$, we cosider the sets $M_1' = M_1 \times \{1\}$, $M_2' = M_2 \times \{2\}$ and we give on the set $M_1' \cup M_2'$ the above monoidoid structure.

This monoidoid is denoted by $M_1 \coprod M_2$ and is called the *disjoint union of monoidoids* M_1 and M_2. Its unit set is $M_0 = M_{1,0} \cup M_{2,0}$, where $M_{i,0}$ is the unit set of M_i, $i = 1,2$.

In particular, the disjoint union of monoids M_i, $i = 1,2$ is a monoidoid, called the *monoidoid associated to monoids* M_i, $i = 1,2$ (for this monoidoid, the unit set is $M_0 = \{e_1, e_2\}$ where e_i is the unity of $M_i, i = 1,2$).

(iv) Let the nul monoid $M_1 = \{e\}$ and the multiplicative monoid $M_2 = \{ -1, 0, 1 \} \subset \mathbf{Z}$. Then $M = M_1 \cup M_2 = \{ e, 1, 0, -1 \}$ is a monoidoid over $M_0 = \{e, 1 \}$. □

Example 1.2. (i) *The monoidoid* $\mathcal{F}(S,X)$. Let X be a given nonepmty set. We denote by $\mathcal{F}(S,X) = \{ f : S \to X \mid (\forall)\ S$ such that $\emptyset \neq S \subseteq X \}$. For $f \in \mathcal{F}(S,X)$, let $D(f)$ be the domain of f and let $R(f) = f(D(f))$. For $M = \mathcal{F}(S,X)$ let $M_{(2)} = \{(f,g) \in M \times M \mid R(f) = D(g) \}$ and for $(f,g) \in M_{(2)}$ define $\mu(f,g) = g \circ f$. If Id_S denotes the identity map on S, then $M_0 = \{ Id_S \mid \emptyset \neq S \subseteq X \}$ is the set of units of M. The maps $\alpha, \beta : M \to M$ are defined by $\alpha(f) = Id_{D(f)}$, $\beta(f) = Id_{R(f)}$. Thus $\mathcal{F}(S,X)$ ia a monoidoid, called the *monoidoid of functions from* S *to* X , where S is an arbitrary nonempty subset of the set X .

(ii) Let Oxy be a system of cartesian coordinates in a plane. We consider the subsets $Ox = \{(x,0) \in \mathbf{R}^2 \mid (\forall)\ x \in \mathbf{R} \}$ and $Oy = \{(0,y) \in \mathbf{R}^2 \mid (\forall)\ y \in \mathbf{R} \}$ of $X = \mathbf{R}^2$. Let $M = \{f_1 = Id_{Ox}, f_2 = Id_{Oy}, f_3 = \sigma_{Ox}, f_4 = \sigma_{Oy} \} \subset \mathcal{F}(S, \mathbf{R}^2)$ where the functions $f_3 : Ox \to Oy$, $f_4 : Oy \to Ox$ are defined by $f_3(x,0) = (0,x)$ and $f_4(0,y) = (y,0)$ (σ_{Ox} resp. σ_{Oy} is called the *saltus function defined on* x *-axis* resp. y *-axis*).

The set of composable pairs of M is

$M_{(2)} = \{ (f_1,f_1); (f_1,f_3); (f_2,f_2); (f_2,f_4); (f_3,f_2); (f_3,f_4); (f_4,f_1); (f_4,f_3) \}$.

We have:

$\mu(f_1,f_1) = f_1 \circ f_1 = f_1 \in M$; $\mu(f_1,f_3) = f_3 \circ f_1 = f_3 \in M$; $\mu(f_2,f_2) = f_2 \circ f_2 = f_2 \in M$;

$\mu(f_2,f_4) = f_4 \circ f_2 = f_4 \in M$; $\mu(f_3,f_2) = f_2 \circ f_3 = f_3 \in M$; $\mu(f_3,f_4) = f_4 \circ f_3 = f_1 \in M$;

$\mu(f_4,f_1) = f_1 \circ f_4 = f_4 \in M$; $\mu(f_4,f_3) = f_3 \circ f_4 = f_2 \in M$.

The unit set of M is $M_0 = \{ f_1 = Id_{Ox}, f_2 = Id_{Oy} \}$. The source and target map $\alpha, \beta : M \to M_0$ are given by

$\alpha(f_1) = \beta(f_1) = f_1$; $\alpha(f_2) = \beta(f_2) = f_2$; $\alpha(f_3) = \beta(f_4) = f_1$; $\alpha(f_4) = \beta(f_3) = f_2$.

We obtain that $(M = \{ f_1, f_2, f_3, f_4 \}, \alpha, \beta; M_0 = \{ f_1, f_2 \})$ is a monoidoid. □

Definition 1.3. Let $(G, \alpha, \beta, \mu; G_0)$ be a monoidoid endowed with

an injective map $\iota : G \rightarrow G,\ x \rightarrow \iota(x)$, called the *inversion map* (we shall write x^{-1} for $\iota(x)$). We say that $(G, \alpha, \beta, \mu, \iota; G_0)$ is a *groupoid*, if the *inverses property* holds, i.e. for each $x \in G$ we have $(x^{-1}, x),\ (x, x^{-1}) \in G_{(2)}$ and $x^{-1}x = \beta(x),\ xx^{-1} = \alpha(x)$. $\square$

The subset $G_0 = \alpha(G) = \beta(G)$ of G is called the *unit set* of G and we say that G is a G_0 - *groupoid*. For all unit $u \in G_0$ we have $\alpha(u) = \beta(u) = \iota(u) = u$.

A G_0 -groupoid G will be denoted by $(G, \alpha, \beta; G_0)$ or $(G; G_0)$. The maps α, β, μ and ι are called the *structure functions* of G. The map $(\alpha, \beta) : G \rightarrow G_0 \times G_0,\ x \rightarrow (\alpha, \beta)(x) = (\alpha(x), \beta(x))$ is called the *anchor map* of the groupoid G. A groupoid $(G, \alpha, \beta; G_0)$ is called *transitive*, if the anchor map $(\alpha, \beta) : G \rightarrow G_0 \times G_0$ is surjective.

By *group bundle* we mean a G_0 -groupoid G such that $\alpha(x) = \beta(x)$ for all $x \in G$. Moreover,a group bundle is the union of its isotropy groups $G(u) = \alpha^{-1}(u), u \in G_0$ (here, two elements may be composed iff they lie in the same fiber $\alpha^{-1}(u)$).

If $(G, \alpha, \beta; G_0)$ is a groupoid then $Is(G) = \{x \in G \mid \alpha(x) = \beta(x)\}$ is a group bundle, called the *isotropy group bundle* of G.

Remark 1.1. (i) The definition of the Brandt groupoid is essentially the same as the one given by A. Coste, P. Dazord and A. Weinstein in [2].

(ii) A groupoid is a monoidoid in which every element is invertible. $\square$

Example 1.3. (i) We consider monoidoid $M = \{\ e, 1, 0, -1\ \}$ over $M_0 = \{e, 1\ \}$, see Example 1.1 (iv). This monoidoid is not a groupoid, since the element 0 is not invertible.

(ii) Let $M = \{f_1 = Id_{Ox}, f_2 = Id_{Oy}, f_3, f_4\ \}$ where $Ox = \{(x, 0) \in \mathbf{R}^2 \mid (\forall)\, x \in \mathbf{R}\ \}$, $Oy = \{(0, y) \in \mathbf{R}^2 \mid (\forall)\, y \in \mathbf{R}\ \}$, $f_3 : Ox \rightarrow Oy,\ (x, 0) \rightarrow f_3(x, 0) = (0, x)$ and $f_4\ : Oy \rightarrow Ox,\ (0, y) \rightarrow f_4(0, y) = (y, 0)$. We have that $(M = \{\ f_1, f_2, f_3, f_4\ \}, \alpha, \beta; M_0 = \{\ f_1, f_2\ \}\ )$ is a monoidoid, see Example 1.2 (ii).

We define the map $\iota : M \rightarrow M$ by taking $\iota(f_1) = f_1,\ \iota(f_2) = f_2,\ \iota(f_3) = f_4$ and $\iota(f_4) = f_3$. It is easy to verify that $(M, \alpha, \beta, \mu, \iota; M_0)$ is a groupoid, called the *groupoid of saltus functions defined on the axes of coordinates in a plane.* We will denote this groupoid by $\mathcal{F}_{(4;2)}(\mathbf{R}^2)$.

The groupoid $(G = \mathcal{F}_{(4;2)}(\mathbf{R}^2) = \{f_1, f_2, f_3, f_4\}, \alpha, \beta, \mu, \iota; G_0 = \{f_1, f_2\})$ is a transitive groupoid. Indeed,
for $(f_i, f_i) \in G_0 \times G_0$ exists $f_i \in G$ such that $\alpha(f_i) = f_i$ and $\beta(f_i) = f_i$ for $i = 1, 2$;
for $(f_1, f_2) \in G_0 \times G_0$ exists $f_3 \in G$ such that $\alpha(f_3) = f_1$ and $\beta(f_3) = f_2$, and
for $(f_2, f_1) \in G_0 \times G_0$ exists $f_4 \in G$ such that $\alpha(f_4) = f_2$ and $\beta(f_4) = f_1$. Hence, the anchor map is surjective. $\square$

In the following proposition we summarize some properties of the structure functions of a groupoid obtained directly from definitions.

Proposition 1.2. *Let* $(G, \alpha, \beta, \mu, \iota; G_0)$ *be a groupoid and* $u \in G_0$.

Then the following assertions hold:
(1.5) $\alpha(x^{-1}) = \beta(x)$, $\beta(x^{-1}) = \alpha(x)$ *and* $(x^{-1})^{-1} = x$, *for all* $x \in G$;
(1.6) *(cancellation law) If* $x \cdot z_1 = x \cdot z_2$ *(resp.,* $z_1 \cdot x = z_2 \cdot x$ *), then* $z_1 = z_2$;
(1.7) $G(u) = \alpha^{-1}(u) \cap \beta^{-1}(u) = \{x \in G \mid \alpha(x) = \beta(x) = u\}$ *is a group under the restriction of* μ *to* $G(u)$, *called the isotropy group at* u *of* G;
(1.8) *If* G *is transitive, then the isotropy groups* $G(u)$, $u \in G_0$ *are isomorphes.*

Proof. Using the Proposition 1.1 (each groupoid is a monoidoid) and the definitions, it is easy to prove that the assertions $(1.5) - (1.8)$ are valid. □

Definition 1.4. Let $(G, \alpha, \beta, \mu, \iota; G_0)$ and $(G', \alpha', \beta', \mu', \iota'; G'_0)$ be two groupoids. A *morphism of groupoids* or *groupoid morphism* from G into G' is a map $f : G \longrightarrow G'$ such that $f(\mu(x, y)) = \mu'(f(x), f(y))$ for all $(x, y) \in G_{(2)}$. □

A morphism of groupoids $f : G \rightarrow G'$ such that the map f is bijective is called *isomorphism of groupoids* or *groupoid isomorphism.*

The category $\mathcal{BG}roid$ of groupoids has as its objects all groupoids $(G, \alpha, \beta; G_0)$ and as morphisms from $(G, \alpha, \beta; G_0)$ to $(G', \alpha', \beta'; G'_0)$ the set of morphisms of groupoids.

Example 1.4. (i) A group $\mathcal{G}$ having e as unit element is just a $\{e\}$ - groupoid and conversely, every groupoid with one unit element is a group. It follows that the category $\mathcal{G}r$ of groups is a subcategory of the category $\mathcal{BG}roid$.

(ii) Any nonempty set G_0 may be regarded as a nul monoidoid on itself (see, Example 1.1 (ii)). If we consider the map $\iota : G_0 \rightarrow G_0$ as the identity map on G_0, we obtain that $(G_0, \alpha = \beta = Id_{G_0}, \mu, \iota = Id_{G_0}; G_0)$ is a groupoid, called the *nul groupoid* associated to set G_0.

(iii) *Disjoint union of two groupoids.* Let $(G_i, \alpha_i, \beta_i, \mu_i, \iota_i; G_{0,i})$, $i = 1, 2$ be two groupoids such that $G_1 \cap G_2 = \emptyset$.

We consider the disjoint union $(G = G_1 \coprod G_2, \alpha, \beta, \mu; G_0 = G_{1,0} \coprod G_{2,0})$ of the monoidoids G_1 and G_2 (see, Example 1.1 (iii)).

We define the map $\iota : G \rightarrow G$ by taking $\iota(x) = \iota_1(x)$, if $x \in G_1$ and $\iota(x) = \iota_2(x)$ if $x \in G_2$. We have that $(G = G_1 \coprod G_2, \alpha, \beta, \mu, \iota; G_0 = G_{1,0} \coprod G_{2,0})$ is a groupoid, called the *disjoint union* of the groupoids G_1 and G_2.

In particular, the disjoint union of groups G_i, $i = 1, 2$ is a groupoid, called the *groupoid associated to groups* G_i, $i = 1, 2$ (for this groupoid, the unit set is $G_0 = \{e_1, e_2\}$ where e_i is the unity of $G_i, i = 1, 2$). □

A finite groupoid $(G; G_0)$ such that $\mid G \mid = n$ and $\mid G_0 \mid = m$ is called $(n; m)-$ *groupoid* or *finite groupoid of type* $(n; m)$.

Example 1.5. (i) Each finite groupoid of type $(n; 1)$ is a group.

(ii) Each finite groupoid of type $(n; n)$ is a nul groupoid.

(iii) The groupoid $\mathcal{F}_{(4;2)}(\mathbf{R}^2)$ (see, Example 1.3 (ii)) is a finite transitive groupoid of type $(4;2)$.

(iv) Let be the Klein 4 -group $K_4 = \{ (1),\ \sigma = (12)(34),\ \tau = (13)(24),\ \sigma \circ \tau = (14)(23) \} \subset S_4$ (it is a subgroup of the symmetric group S_4 of degree 4). We have $\sigma^2 = \tau^2 = (1)$ and $\tau \circ \sigma = \sigma \circ \tau$.

We consider the disjoint union $G = K_4 \coprod \mathbf{Z}_4$ of the groups K_4 and $\mathbf{Z}_4$, where $\mathbf{Z}_4 = \{ \widehat{0}, \widehat{1}, \widehat{2}, \widehat{(3)} \}$ is the group of congruences classes of integers modulo 4. We obtain a groupoid G of order 8 with unit set $G_0 = \{ (1), \widehat{0} \}$ of type $(8;2)$.

The structure functions α, β , ι and μ of $G = K_4 \coprod \mathbf{Z}_4$ are given in the following tables:

x	(1)	σ	τ	$\sigma \circ \tau$	$\widehat{0}$	$\widehat{1}$	$\widehat{2}$	$\widehat{3}$
$\alpha(x)$	(1)	(1)	(1)	(1)	$\widehat{0}$	$\widehat{0}$	$\widehat{0}$	$\widehat{0}$
$\beta(x)$	(1)	(1)	(1)	(1)	$\widehat{0}$	$\widehat{0}$	$\widehat{0}$	$\widehat{0}$
$\iota(x)$	(1)	σ	τ	$\sigma \circ \tau$	$\widehat{0}$	$\widehat{3}$	$\widehat{2}$	$\widehat{1}$

μ	(1)	σ	τ	$\sigma \circ \tau$	$\widehat{0}$	$\widehat{1}$	$\widehat{2}$	$\widehat{3}$
(1)	(1)	σ	τ	$\sigma \circ \tau$				
σ	σ	(1)	$\tau \circ \sigma$	τ				
τ	τ	$\sigma \circ \tau$	(1)	σ				
$\sigma \circ \tau$	$\sigma \circ \tau$	τ	σ	(1)				
$\widehat{0}$					$\widehat{0}$	$\widehat{1}$	$\widehat{2}$	$\widehat{3}$
$\widehat{1}$					$\widehat{1}$	$\widehat{2}$	$\widehat{3}$	$\widehat{0}$
$\widehat{2}$					$\widehat{2}$	$\widehat{3}$	$\widehat{0}$	$\widehat{1}$
$\widehat{3}$					$\widehat{3}$	$\widehat{0}$	$\widehat{1}$	$\widehat{2}$

$\square$

2. The program $BGroidAP1$ for to test if an universal algebra is a groupoid

We consider a given finite universal algebra $(G, \alpha, \beta, \mu, \iota; G_0)$ such that $\mid G \mid\ = n$ and $\mid G_0 \mid\ = m$ with $1 \leq m \leq n$. We denote the elements of G by $a_1, a_2, \cdots, a_m, a_{m+1}, \cdots, a_n$ such that $G_0 = \{ a_1, a_2, \cdots, a_m \}$. Hence, the elements a_k, $k = \overline{1, m}$ are the units of G.

We give an algorithm for decide if the universal algebra $(G, \alpha, \beta, \mu, \iota; G_0)$ is a G_0 - groupoid. This algorithm is constituted by the following stages.

Stage I. *We introduce the initial data*: n - the number of elements of G; m - the number of elements of G_0; the functions α, β, ι and μ given by its tables of structure.

Stage II. *Test if the universal algebra* $(G, \alpha, \beta, \mu, \iota; G_0)$ *considered in the first stage is a groupoid*. For this, the following steps are executed:

step 1. $(G, \alpha, \beta, \mu, \iota)$ is a structure well-defined, i.e. the functions α, β

are surjections, ι is injective and μ is defined on the composable pairs $G_{(2)}$ with values in G;
step 2. (G, α, β, μ) is a semigroupoid, i.e. the multiplication law μ is associative;
step 3. the semigroupoid $(G; G_0)$ is a monoidoid, i.e. the identities property is verified;
step 4. the monoidoid $(G; G_0)$ is a groupoid, i.e. each element of G is invertible.
step 5. If the above steps are satisfied, make the tables of the structure functions α, β, ι and μ and write the mesage "*G is a groupoid*".

Let us we present the correspondence between the initial data and input data:

$$G = \{\ a_1, a_2, \ldots, \ldots, a_m, a_{m+1}, \ldots, a_n\ \} \longleftrightarrow \{\ 1, 2, \ldots, m, m+1, \ldots, n\ \}$$

$$\textit{Initial data} \longleftrightarrow \textit{Input data}$$

$$\mid G \mid = n \longleftrightarrow n$$

$$\mid G_0 \mid = m \longleftrightarrow m$$

a_k	a_1	$\cdots$	a_m	a_{m+1}	$\cdots$	a_n
$\alpha(a_k)$	a_1	$\cdots$	a_m	$\alpha(a_{m+1})$	$\cdots$	$\alpha(a_n)$
$\beta(a_k)$	a_1	$\cdots$	a_m	$\beta(a_{m+1})$	$\cdots$	$\beta(a_n)$
$\iota(a_k)$	a_1	$\cdots$	a_m	$\iota(a_{m+1})$	$\cdots$	$\iota(a_n)$

$$\longleftrightarrow \begin{array}{llll} 1 \cdots & m & u_l(m+1) \ \cdots & u_l(n) \\ 1 \cdots & m & u_r(m+1) \ \cdots & u_r(n) \\ 1 \cdots & m & inv(m+1) \cdots & inv(n) \end{array}$$

μ	a_1	$\cdots$	a_k	$\cdots$	a_n
a_1					
$\cdots$					
a_j			$a_j \cdot a_k$		
$\cdots$					
a_n					

$$\longleftrightarrow \begin{array}{ccccc} a_{11} & \cdots & a_{1k} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2k} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{j1} & \cdots & a_{jk} & \cdots & a_{jn} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & \cdots & a_{nk} & \cdots & a_{nn}. \end{array}$$

In the table of μ the element $\mu(a_j, a_k) = a_j \cdot a_k$ is defined iff $\beta(a_j) = \alpha(a_k)$. The absence of an element from the arrow "j" and the column "k" indicates the fact that the pair $(a_j, a_k) \in G \times G$ is not composable. The element a_{jk} is represented by 0 if the product $a_j \cdot a_k$ is not defined.

Example 2.1. Consider the groupoid $G = \mathcal{F}_{(4;2)}\mathbf{R}^2$ given in Example 1.3 (ii). We have

$$G = \{\ f_1 = Id_{Ox}, f_2 = Id_{Oy}, f_3 = \sigma_{Ox}, f_4 = \sigma_{Oy}\ \} \longleftrightarrow \{\ 1, 2, 3, 4\ \}$$

$$\textit{Initial data} \longleftrightarrow \textit{Input data}$$

$$\mid G \mid = 4 \longleftrightarrow 4$$

$$\mid G_0 \mid = 2 \longleftrightarrow 2$$

f_k	f_1	f_2	f_3	f_4
$\alpha(f_k)$	f_1	f_2	f_1	f_2
$\beta(f_k)$	f_1	f_2	f_2	f_1
$\iota(f_k)$	f_1	f_2	f_4	f_3

$$\longleftrightarrow \quad \begin{matrix} 1 & 2 & 1 & 2 \\ 1 & 2 & 2 & 1 \\ 1 & 2 & 4 & 3 \end{matrix}$$

μ	f_1	f_2	f_3	f_4
f_1	f_1		f_3	
f_2		f_2		f_4
f_3		f_3		f_1
f_4	f_4		f_2	

$$\longleftrightarrow \quad \begin{matrix} 1 & 0 & 3 & 0 \\ 0 & 2 & 0 & 4 \\ 0 & 3 & 0 & 1 \\ 4 & 0 & 2 & 0 \end{matrix}$$

□

The implementation of the above algorithm on computer is realized in the program $BGroidAP1$. This program is composed from two modules denoted by $unit11.dfm$ and $unit11.pas$. The module $unit11.pas$ is composed from the principal program followed of procedures and functions. This is presented as follows.

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, DBGrids, ShellAPI, Db, DBTables, StdCtrls, Menus, ExtCtrls,
  ComCtrls, ToolWin, Spin;

const
      nmax = 200;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    OpenFile1: TMenuItem;
    SaveFile1: TMenuItem;
    GroupBox1: TGroupBox;
    StringGrid1: TStringGrid;
    StringGrid2: TStringGrid;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    ToolBar1: TToolBar;
    Splitter4: TSplitter;
    ToolButton1: TToolButton;
```

```
    New1: TMenuItem;
    ToolBar3: TToolBar;
    ToolButton4: TToolButton;
    ToolButton5: TToolButton;
    Label2: TLabel;
    SpinEdit1: TSpinEdit;
    ToolButton6: TToolButton;
    Label3: TLabel;
    SpinEdit2: TSpinEdit;
    StatusBar1: TStatusBar;
    ToolButton3: TToolButton;
    ToolButton11: TToolButton;
    procedure FormShow(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure StringGrid1SetEditText(Sender: TObject; ACol, ARow:
Integer;
      const Value: String);
    procedure StringGrid2SetEditText(Sender: TObject; ACol, ARow:
Integer;
      const Value: String);
    procedure OpenFile1Click(Sender: TObject);
    procedure SaveFile1Click(Sender: TObject);
    procedure New1Click(Sender: TObject);
    procedure ToolButton4Click(Sender: TObject);
    procedure ToolButton3Click(Sender: TObject);
  private
    err_message : String;
        m, n : Integer;
    h : array[0..nmax, 0..nmax] of Byte;
    u_left, u_right, inv : array[0..nmax] of Integer;

    procedure WMDropFiles(var Msg: TWMDropFiles); message WM_DROPFILES;
    procedure PerformFileOpen(const FileName1 : string);
    procedure PerformFileSave(const FileName1 : string);
    procedure MakeUnitsTable;
              procedure MakeGroupoidTable;
    function ToStr(x : Integer) : String;
    function IsStructure : Boolean;
    function IsSemigroupoid : Boolean;
    function IsMonoidoid : Boolean;
    function IsGroupoid : Boolean;
  public

  end;
```

```
var
   Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.FormShow(Sender: TObject);
var
         i, j : Byte;
begin
   DragAcceptFiles(Handle, True);
   StringGrid1.EditorMode := True;
   n := 0;
   m := 0;
   for i := 0 to nmax do
         for j := 0 to nmax do
         h[i, j] := 0;
   for i := 0 to nmax do begin
     u_left[i] := 0;
     u_right[i] := 0;
     inv[i] := 0;
   end;
end;

function TForm1.ToStr;
var
         ss : String;
begin
   str(x, ss);
   ToStr := ss;
end;

procedure TForm1.MakeUnitsTable;
var
         i : Integer;
begin
         StringGrid2.RowCount := 4;
         StringGrid2.ColCount := n + 1;
         StringGrid2.Cells[0, 1] := 'u_l';
         StringGrid2.Cells[0, 2] := 'u_r';
         StringGrid2.Cells[0, 3] := 'inv';
   for i := 1 to n do begin
         StringGrid2.Cells[i, 0] := tostr(i);
         StringGrid2.Cells[i, 1] := tostr(u_left[i]);
         StringGrid2.Cells[i, 2] := tostr(u_right[i]);
```

```
        StringGrid2.Cells[i, 3] := tostr(inv[i])
    end
end;

procedure TForm1.MakeGroupoidTable;
var
    i, j : Integer;
begin
    StatusBar1.SimpleText := 'G has not been tested';
    if n > 0 then begin
      StringGrid1.RowCount := n + 1;
      StringGrid1.ColCount := n + 1;
      for i := 1 to n do begin
        StringGrid1.Cells[0, i] := tostr(i);
        StringGrid1.Cells[i, 0] := tostr(i);
      end;
      for i := 1 to n do
        for j := 1 to n do
          if h[i, j] <> 0 then
            StringGrid1.Cells[j, i] := tostr(h[i, j])
          else
            StringGrid1.Cells[j, i] := ''
    end else begin
      StringGrid1.RowCount := 2;
      StringGrid1.ColCount := 2
    end;
    MakeUnitsTable
end;

function TForm1.IsStructure;
var
        i, j : Byte;
begin
        IsStructure := true;
    for i := 1 to n do begin
        if (u_left[i] = 0) or (u_right[i] = 0) or (inv[i] = 0) then begin
                IsStructure := false;
        err_message := 'Structure incomplete';
        exit
      end
    end;
    for i := 1 to n do for j := 1 to n do
        if (u_right[i] = u_left[j]) and (h[i, j] = 0) then begin
                IsStructure := false;
```

```
      err_message := 'Structure incomplete';
      exit
     end
end;

function TForm1.IsSemigroupoid;
var
         i, j, k : Byte;
begin
         if IsStructure then begin
     IsSemigroupoid := true;
     for i := 1 to n do for j := 1 to n do if u_right[i] = u_left[j] then
      for k := 1 to n do if u_right[j] = u_left[k] then
         if h[h[i, j], k] <> h[i, h[j, k]] then begin
           IsSemigroupoid := false;
           err_message := tostr(i) + ', ' + tostr(j) + ', ' + tostr(k) + ' -
not asociative';
           exit
         end
   end else IsSemigroupoid := false
end;

function TForm1.IsMonoidoid;
var
         i : Byte;
begin
         if IsSemigroupoid then begin
     IsMonoidoid := true;
     for i := 1 to n do
      if (h[u_left[i], i] <> i) or (h[i, u_right[i]] <> i) then begin
         IsMonoidoid := false;
         err_message := tostr(i) + ' has no unit';
         exit
      end
   end else IsMonoidoid := false
end;

function TForm1.IsGroupoid;
var
         i : Byte;
begin
         if IsMonoidoid then begin
     IsGroupoid := true;
     for i := 1 to n do begin
      if u_right[i] = u_left[inv[i]] then
```

```
        if h[i, inv[i]] <> u_left[i] then begin
          IsGroupoid := false;
          err_message := tostr(i) + ' has no inverse';
          exit
        end;
      if u_right[inv[i]] = u_left[i] then
        if h[inv[i], i] <> u_right[i] then begin
          IsGroupoid := false;
          err_message := tostr(i) + ' - has no inverse';
          exit
        end
    end
  end else IsGroupoid := false
end;

procedure TForm1.WMDropFiles(var Msg: TWMDropFiles);
var
  CFileName: array[0..MAX_PATH] of Char;
begin
  try
    if DragQueryFile(Msg.Drop, 0, CFileName, MAX_PATH) > 0 then
    begin
     {CheckFileSave;}
     PerformFileOpen(CFileName);
     Msg.Result := 0;
    end;
  finally
    DragFinish(Msg.Drop);
  end;
end;

procedure TForm1.PerformFileOpen(const FileName1 : string);
var
        fin : TextFile;
  i, j : Integer;
begin
        AssignFile(fin, FileName1);
  reset(fin);
  readln(fin, n);
  readln(fin, m);
  readln(fin);
  for i := 1 to n do
        read(fin, u_left[i]);
  for i := 1 to n do
```

```
        read(fin, u_right[i]);
   for i := 1 to n do
        read(fin, inv[i]);
   for i := 1 to n do
        for j := 1 to n do
        read(fin, h[i, j]);
   CloseFile(fin);
   MakeGroupoidTable;
end;

procedure TForm1.PerformFileSave(const FileName1 : string);
var
        f : TextFile;
   i, j : Integer;
begin
        AssignFile(f, FileName1);
   rewrite(f);
   writeln(f, n);
   writeln(f, m);
   writeln(f);
   for i := 1 to n do
        write(f, u_left[i], ' ');
   writeln(f);
   for i := 1 to n do
        write(f, u_right[i], ' ');
   writeln(f);
   for i := 1 to n do
        write(f, inv[i], ' ');
   writeln(f);
   writeln(f);
   for i := 1 to n do begin
        for j := 1 to n do
        write(f, h[i, j], ' ');
     writeln(f);
   end;
   CloseFile(f)
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
        MakeGroupoidTable;
        err_message := '';
        if IsGroupoid then
        StatusBar1.SimpleText := 'G is a groupoid'
```

```
    else
           StatusBar1.SimpleText := err_message;
end;

procedure TForm1.StringGrid1SetEditText(Sender: TObject; ACol,
    ARow: Integer; const Value: String);
var
           nr : Byte;
    cod : Integer;
begin
           Val(Value, nr, cod);
    if cod <> 0 then
           h[ARow, ACol] := 0
    else
                  h[ARow, ACol] := nr
end;

procedure TForm1.StringGrid2SetEditText(Sender: TObject; ACol,
    ARow: Integer; const Value: String);
var
           nr : Byte;
    cod : Integer;
begin
    Val(Value, nr, cod);
    if cod <> 0 then nr := 0;
           case ARow of
           1 : u_left[ACol] := nr;
           2 : u_right[ACol] := nr;
           3 : inv[ACol] := nr
    end
end;

procedure TForm1.OpenFile1Click(Sender: TObject);
begin
           if OpenDialog1.Execute then
                  PerformFileOpen(OpenDialog1.FileName)
end;

procedure TForm1.SaveFile1Click(Sender: TObject);
begin
           if SaveDialog1.Execute then
           PerformFileSave(SaveDialog1.FileName)
end;

procedure TForm1.New1Click(Sender: TObject);
begin
```

```
    SpinEdit1.Value := n;
    SpinEdit2.Value := m;
          ToolBar3.Show;
    ToolBar1.Hide
end;

procedure TForm1.ToolButton3Click(Sender: TObject);
begin
          ToolBar1.Show;
          ToolBar3.Hide
end;

procedure TForm1.ToolButton4Click(Sender: TObject);
var
          i, j : Byte;
begin
          n := SpinEdit1.Value;
          m := SpinEdit2.Value;
    for i := 1 to n do
          for j := 1 to n do
          h[i, j] := 0;
    ToolBar1.Show;
    ToolBar3.Hide;
    MakeGroupoidTable
end;
end.
```

We illustrate the utilisation of the program $BGgroidAP1$ in the following examples.

Example 2.2. Consider the universal algebra $(G, \alpha, \beta, \mu, \iota; G_0)$, where $G = \{ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9 \}$, $G_0 = \{ x_1, x_2, x_3 \}$ and the inputs data are the following:

```
9
3
1 2 3 1 1 2 2 3 3
1 2 3 2 3 1 3 1 2
1 2 3 6 8 4 9 5 7
```

```
1 0 0 4 5 0 0 0 0
0 2 0 0 0 6 7 0 0
0 0 3 0 0 0 0 8 9
0 4 0 0 0 1 5 0 0
0 0 5 0 0 0 0 1 4
6 0 0 2 7 0 0 0 0
0 0 7 0 0 0 0 6 2
8 0 0 9 3 0 0 0 0
0 9 0 0 0 8 3 0 0
```

Execute the program $BGroidAP1$ for $G = \{ \ x_j \mid j = \overline{1,9} \ \}$ and the window program of obtained results is presented in the Figure 1.

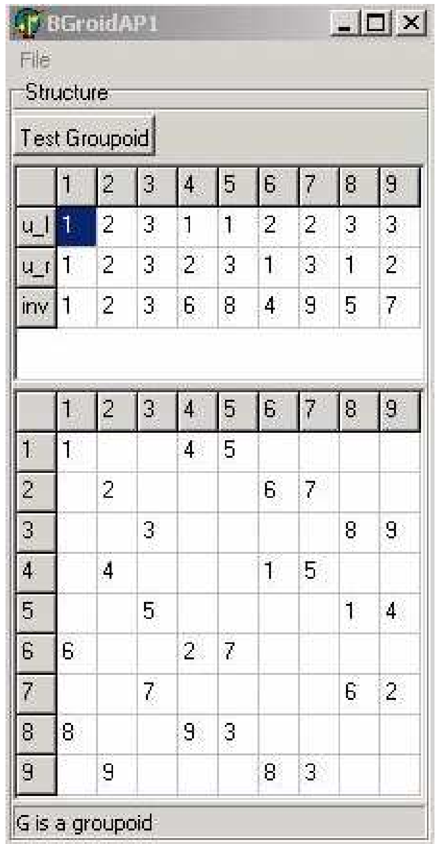

Figure 1: A $(9;3)$ - groupoid

Therefore, G is a groupoid of type $(9;3)$. $\square$

Example 2.3. Consider the universal algebra $(G, \alpha, \beta, \mu, \iota; G_0)$, where $G = \{ \ a_1, a_2, a_3, a_4, a_5, a_6 \ \}$, $G_0 = \{ \ a_1 \ \}$ and the inputs data are the following:

$$\begin{array}{cccccc} 6 & & & & & \\ 1 & & & & & \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 2 & 4 & 6 & 5 \end{array} \qquad \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 3 & 1 & 6 & 4 & 5 \\ 3 & 1 & 2 & 5 & 6 & 4 \\ 4 & 5 & 6 & 1 & 2 & 3 \\ 5 & 6 & 4 & 3 & 1 & 2 \\ 6 & 4 & 5 & 2 & 3 & 1 \end{array}$$

$\square$

Execute the program $BGroidAP1$ for $G = \{ \ a_k \mid k = \overline{1,6} \ \}$ and the window program of obtained results is presented in the Figure 2.

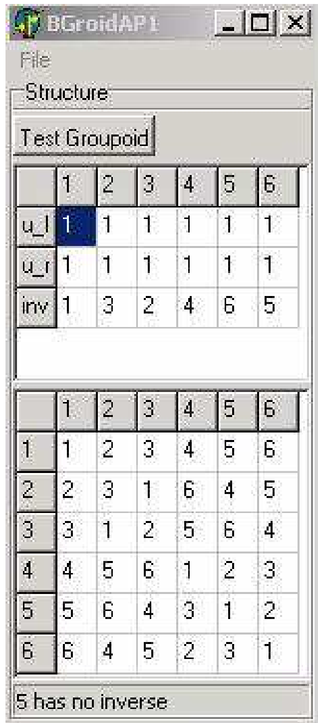

Figure 2: A monoid which is not a group

Therefore, G is not a groupoid of type $(6;1)$ (hence , G is not a group). We have that G is a monoid. $\square$

Theorem 2.1. (**theorem of classification**) *Let* $(G, \alpha, \beta, \mu, \iota; G_0)$ *be a groupoid of type* $(4;2)$. *Then the groupoid* G *is isomorphic with one from the following groupoids:*

(i) $G \cong \{ e \} \coprod \mathbf{Z}_3$; (ii) $G \cong \mathbf{Z}_2 \coprod \mathbf{Z}_2$; (iii) $G \cong \mathcal{F}_{(4;2)}(\mathbf{R}^2)$

where $\mathcal{F}_{(4;2)}(\mathbf{R}^2)$ *is the groupoid of saltus functions defined on the axes of coordinates in a plane.*

Proof. Let the groupoid $(G, \alpha, \beta, \mu, \iota; G_0)$ of type $(4;2)$, where $G = \{ a_1, a_2, a_3, a_4 \}$ and $G_0 = \{ a_1, a_2 \}$.

For the surjections $\alpha, \beta : G \rightarrow G_0$ with properties $\alpha(a_k) = \beta(a_k) = a_k$, for $k = 1, 2$ exists the following situations given by the tables :

Case1.

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_2	a_2
$\beta(a)$	a_1	a_2	a_2	a_2

Case2.

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_1	a_2
$\beta(a)$	a_1	a_2	a_1	a_2

Case3.

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_1	a_2
$\beta(a)$	a_1	a_2	a_2	a_1

In the **Case 1**, for the injective map $\iota : G \rightarrow G$ with property $\iota(a_k) = a_k$, for $k = 1, 2$ exists the following cases given by the tables:

Case1.1.

a	a_1	a_2	a_3	a_4
$\iota(a)$	a_1	a_2	a_3	a_4

Case1.2.

a	a_1	a_2	a_3	a_4
$\iota(a)$	a_1	a_2	a_4	a_3

In the **Case 1** the set $G_{(2)}$ of composable pairs of the groupoid $(G; G_0)$ is $G_{(2)} = \{(a_1, a_1); (a_2, a_2); (a_2, a_3); (a_2, a_4); (a_3, a_2); (a_3, a_3); (a_3, a_4); (a_4, a_2); (a_4, a_3); (a_4, a_4)\}$.

Case 1.1. Using the fact that $\mu(a_k, a_k) = a_k$ for $k = 1, 2$ and the properties of the functions α, β and ι , the structure table of μ in the situation of **Case 1.1** is the following :

μ	a_1	a_2	a_3	a_4
a_1	a_1			
a_2		a_2	a_3	a_4
a_3		a_3	a_4	$\mu(a_3, a_4)$
a_4		a_4	$\mu(a_4, a_3)$	a_3

In this case, it follows that for μ we have the following situations: **Cases 1.1.1 - 1.1.9** obtained by taking $\mu(a_3, a_4) \in \{ a_2, a_3, a_4 \}$ and $\mu(a_4, a_3) \in \{ a_2, a_3, a_4 \}$.

If we introduce the initial data in each situation of **Case 1.1.1-1.1.9** and apply the program $BGroidAP1$, we obtain that $(G; G_0)$ is not a groupoid.

Case 1.2. Using the fact that $\mu(a_k, a_k) = a_k$ for $k = 1, 2$ and the properties of the functions α, β and ι , the structure table of μ in the situation of **Case 1.2** is the following :

μ	a_1	a_2	a_3	a_4
a_1	a_1			
a_2		a_2	a_3	a_4
a_3		a_3	$\mu(a_3, a_3)$	a_2
a_4		a_4	a_2	$\mu(a_4, a_4)$

In this case, it follows that for μ we have the following situations: **Cases 1.2.1 - 1.2.9** obtained by taking $\mu(a_3, a_3) \in \{ a_2, a_3, a_4 \}$ and $\mu(a_4, a_4) \in \{ a_2, a_3, a_4 \}$.

If we introduce the initial data in each situation of **Case 1.1.1-1.1.9** and apply the program $BGroidAP1$, we obtain that $(G; G_0)$ is a groupoid when $\mu(a_3, a_3) = a_4$ and $\mu(a_4, a_4) = a_3$ (in the other cases, G is not a groupoid). For this situation, the structure functions α, β, ι and μ are given by the tables:

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_2	a_2
$\beta(a)$	a_1	a_2	a_2	a_2
$\iota(a)$	a_1	a_2	a_4	a_3

μ	a_1	a_2	a_3	a_4
a_1	a_1			
a_2		a_2	a_3	a_4
a_3		a_3	a_4	a_2
a_4		a_4	a_2	a_3

It is easy to prove that this groupoid is the union of two groups and it is isomorphic with $\{ e \} \coprod \mathbf{Z}_3$. Hence, $G \cong \{ e \} \coprod \mathbf{Z}_3$.

Case 2. Applying the same methode as in the **Case 1**, we obtain that the groupoid G is the disjoint union $G_1 \cup G_2$, where $G_1 = \{ a_1, a_3 \}$ such that $a_3^2 = a_1$ and $G_2 = \{ a_2, a_4 \}$ such that $a_4^2 = a_2$.

For this groupoid, the structure functions α, β, ι and μ are given by the tables:

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_1	a_2
$\beta(a)$	a_1	a_2	a_1	a_2
$\iota(a)$	a_1	a_2	a_3	a_4

μ	a_1	a_2	a_3	a_4
a_1	a_1		a_3	
a_2		a_2		a_4
a_3	a_3		a_1	
a_4		a_4		a_2

It is easy to prove that this groupoid isomorphic with $\mathbf{Z}_2 \coprod \mathbf{Z}_2$. Hence, $G \cong \mathbf{Z}_2 \coprod \mathbf{Z}_2$.

Case 3. Similarly, we obtain that the structure functions of the groupoid G are given by the tables:

a	a_1	a_2	a_3	a_4
$\alpha(a)$	a_1	a_2	a_1	a_2
$\beta(a)$	a_1	a_2	a_2	a_1
$\iota(a)$	a_1	a_2	a_4	a_3

μ	a_1	a_2	a_3	a_4
a_1	a_1		a_3	
a_2		a_2		a_4
a_3		a_3		a_1
a_4	a_4		a_2	

It is easy to prove that this groupoid isomorphic with $\mathcal{F}_{(4;2)}(\mathbf{R}^2)$, see Example . Hence, $G \cong \mathcal{F}_{(4;2)}(\mathbf{R}^2)$. We observe that in this case , G is not a group bundle. □

For more details concerning the program $BGroidAP1$, the reader can be inform at e-mail adress: ivan@hilbert.math.uvt.ro.

References

[1]. R. Brown, *From Groups to Groupoids : a brief survey.* Bull. London Math. Soc.,**19**, 1987, 113-134.

[2]. A. Coste, P. Dazord and A. Weinstein, *Groupoides symplectiques.* Publ. Dept. Math. Lyon, 2/A, 1987,1-62.

[3]. P. J. Higgins, *Notes on categories and groupoids.* Von Nostrand Reinhold, London, 1971.

[4]. Gh. Ivan, *Cayley Theorem for Monoidoids.* Glasnick Matematicki, Vol. 31(51), 1996, 73-82.

[5]. Gh. Ivan, *Algebraic constructions of Brandt groupoids.* Proceedings of the Algebra Symposium, " Babeş- Bolyai" University, Cluj, 2002, 69-90.

[6]. M. V. Karasev, *Analogues of objects of Lie groups theory for non-linear Poisson brackets.* Math. U.S.S.R. Izv., 26, 1987, 497-527.

[7]. K. Mackenzie, *Lie groupoids and Lie algebroids in differential geometry.* London Math. Soc., Lectures Notes Series, **124**, Cambridge Univ.Press, 1987.

[8]. A. Weinstein, *Groupoids: Unifying Internal and External Symmetries.* Notices Amer. Math. Soc., **43**, 1996, 744 - 752.

Author's address:

West University of Timişoara
Department of Mathematics
4, Bd. V. Pârvan, 1900, Timişoara , Romania
E-mail : ivan@hilbert.math.uvt.ro

West University of Timişoara
Department of Mathematics. Seminarul de Algebră
4, Bd. V. Pârvan, 1900, Timişoara, Romania